Napredni industrijski komunikacioni protokoli,

Primenjeno softversko inženjerstvo,

Fakultet tehničkih nauka, Novi Sad, 2016

Dokumentacija izrade projektnog zadatka - PubSub servis

Opis problema

Dizajn programskog rešenja

Korišćene strukture podataka

Rezultati testiranja

Mogućnosti unapređenja rešenja

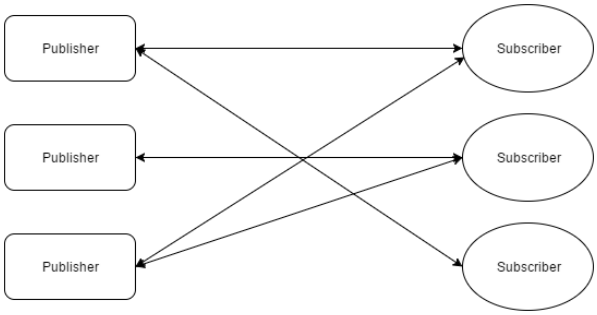
Srđan Paunović E38/2016

Marko Bogdanović E34/2016

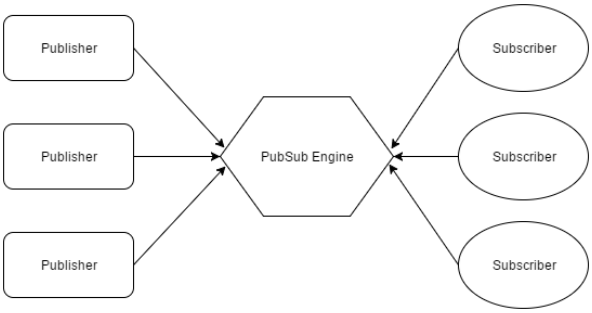
Opis problema

Problemi komunikacije, između učesnika jednog komunikacionog sistema, često su veoma složeni problemi. Neki od faktora koji utiču na projektovanje jednog takvog sistema, mogu biti broj i podela učesnika sistema po ulozi koju obavljaju u istom.

Pretpostavimo komunikacioni sistem sa dve vrste učesnika, oglašivač (eng. publisher) i pretplatnik (eng. subscriber). Misija ovog komunikacionog sistema jeste da informacija koja se tiče određene teme (eng. topic), stigne do svih pretplatnika na tu temu. Međutim, sa oglašivačima i pretplatnicima kao jedinim vrstama učesnika u sistemu, njihova odgovornost narasta srazmerno njihovom broju. S toga, rešenje prethodno opisanog problema, kao i predmet izrade ovog projekta, jeste uvođenje nove komponente u komunikacioni sistem, sa ciljem izbegavanja prevelike odgovornosti oglašivača i pretplatnika. Dakle, uvodimo posrednika u komunikaciji sa dvojakom ulogom. S jedne strane, posrednik dobavlja informaciju primljenu od oglašivača na određenu temu, do svih pretplatnika zainteresovanih za istu. S druge strane, posrednik prima zahteve od pretplatnika i vodi evidenciju o njihovim pretplatama.



Slika 1. Veze između oglašivača i pretplatnika bez posrednika



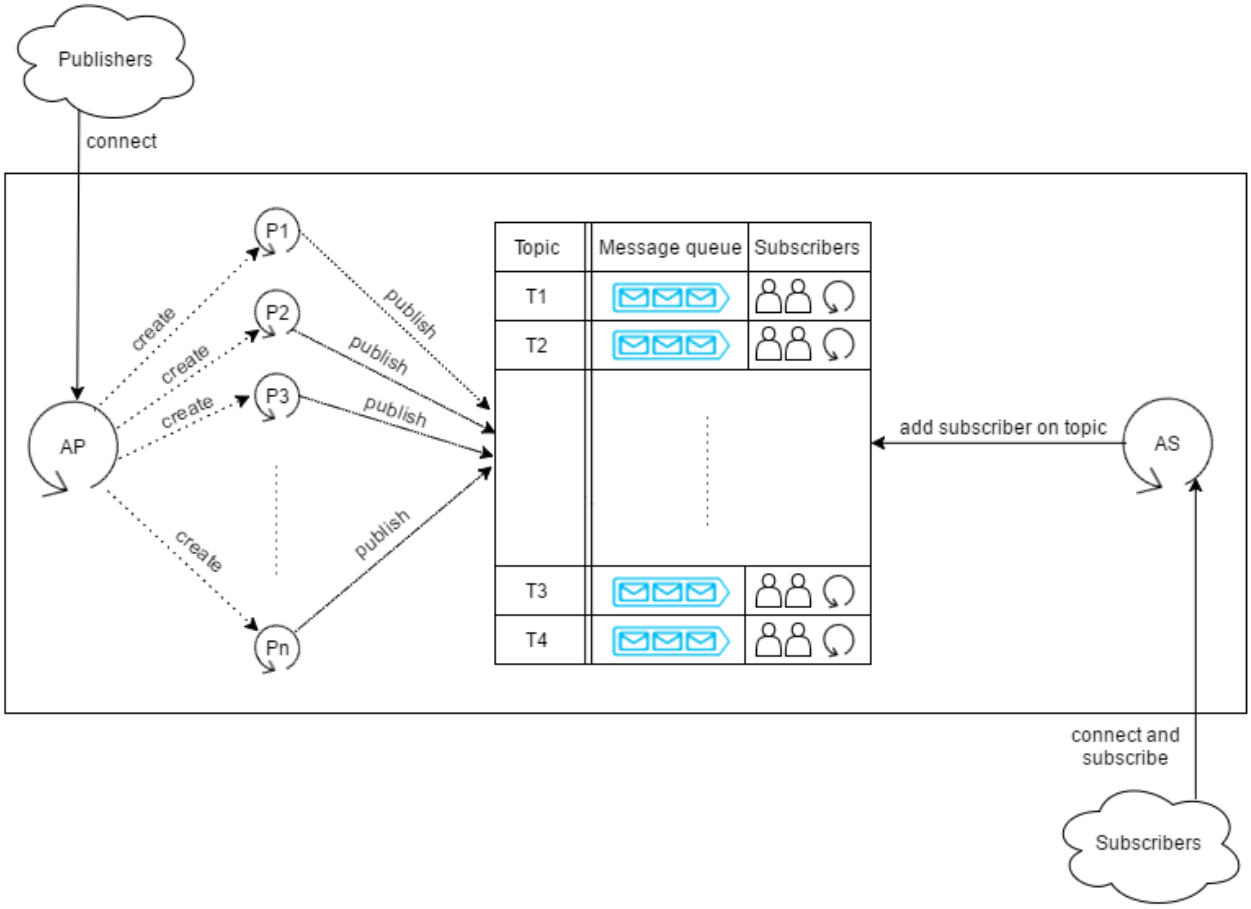
Slika 1. Veze između oglašivača i pretplatnika sa posrednikom

Ovako projektovan sistem obezbeđuje nepromenljivu raspodelu odgovornosti bez obzira na broj obe vrste učesnika i kao takav, ima čestu primenu u različitim komunikacionim sistemima.

Dizajn programskog rešenja

U ovom radu, PubSub servis je implementiran kao višenitni proces. Svakoj niti u okviru procesa, dodeljen je deo odgovornosti za koju je nadležna, a sve zajedno obezbeđuju dobavljanje informacija do odgovarajućih pretplatnika.

Na dijagramu 1, prikazan je dizajn rešenja sa raspodelom odgovornosti po nitima, i internom tabelom koja čuva informacije potrebne za njihovu kooperaciju.



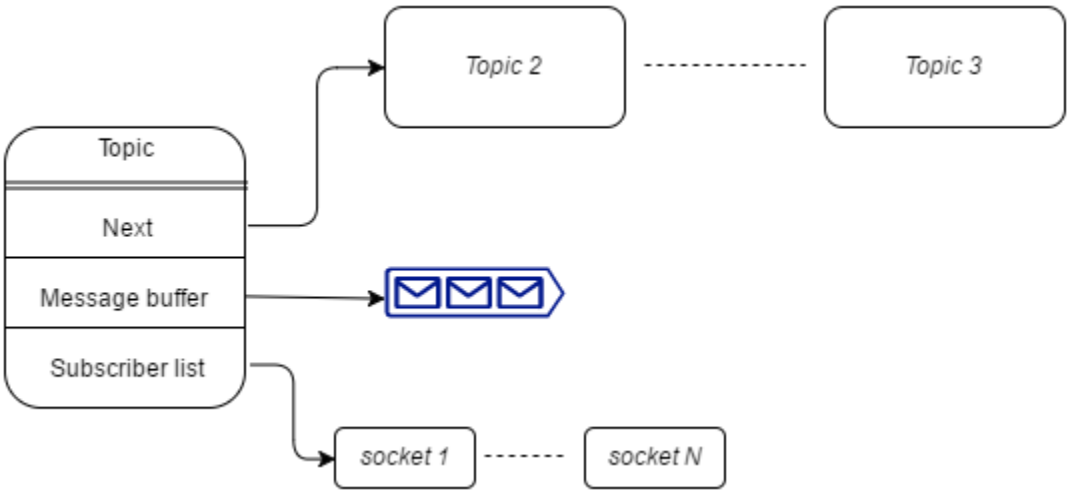
Dijagram 3. Dizajn višenitnog rešenja PubSub servisa

Nit AP obavlja neprestano osluškivanje dolazećih konekcija od strane ograšivača. Pri dolasku novog oglašivača, nit AP kreira novu nit, Pn, čije je zaduženje prihvatanje poruka od priključenog oglašivača. Životni vek novonastale niti prekida se prekidanjem konekcije sa oglašivačem za koju je zadužena. Pri dobijanju nove poruke, nit Pn, reaguje ubacivanjem iste u kružni bafer za određenu temu, na koju je poruka poslata. Svakoj temi, odgovovara jedan kružni bafer za pristigle poruke i spisak pretplatnika na tu temu za čije je obaveštavanje zadužena posebna nit. Ta posebna nit, obaveštava sve pretplatnike na jednu temu pri upisu nove poruke u kružni bafer iste teme. Dodavanje pretplatnika u spisak, vrši nit AS, koja osluškuje nadolazeće pretplatnike, koji šalju skup tema za koje se interesuju.

Korišćene strukture podataka

Svi podaci neophodni za funkcionisanje PubSub servisa, čuvaju se u njegovoj internoj memoriji. Podaci, neophodni za dobavljanje odgovarajućih poruka do odgovarajućeg skupa pretplatnika, organizovani su u logičku tabelu, gde je svaki red jednoznačno identifikovan temom, i koji sadrži red pristiglih poruka zajedno sa skupom pretplatnika. Takođe, servis rukuje i deskriptorima svih aktivnih niti, koje skladišti u skup, koji dinamički ažurira pri kreiranju, odnosno gašenju niti.

Radi implementacije svih prethodno spomenutih logičkih struktura podataka, korišćena je jednostruko spregnuta lista, dok je za skladištenje poruka upotrebljen kružni bafer sa mogućnošću dinamičkog proširivanja.



Dijagram 4. Tabela kao logička struktura podataka implementirana u vidu liste

Rezultati testiranja

Kako bismo izmerili performanse postojećeg rešenja, izvršili smo dve vrste testa. U sklopu obe vrste testa, merili smo procesorsko opterećenje, memorijsko zauzeće, mrežno opterećenje i broj niti koje su aktivne pri datim testovima. Prvi test podrazumevao je slanje 1GB podataka u paketima po 1KB, od jednog oglašivača do jednog pretplatnika. Dok je drugi scenario bio takođe slanje 1GB podataka u paketima po 1KB, ali od jednog oglašivača na deset različitih tema, za koje postoji po jedan pretplatnik. U tabeli 1. dati su rezultati testova.

Test	Procesor [%]	Memorija [b]	Mreža [kb/s]	Broj niti
1.	~12%	~2.500.000	~240	6
2.	~85%	~1.000.000	~170	17

Tabela 1. Rezultati testova

Iz priloženih rezultata može se primetiti da se opterećenje procesora drastično povećalo sa povećanjem broja niti. Bitno je naglasiti, da je povećanje broja niti, posledica povećanog broja tema u okviru servisa, a ne povećanog broja pretplatnika.

Mogućnosti unapređenja rešenja

Nijedno rešenje nije apsolutno optimalno. Prethodno opisani, višenitni dizajn PubSub servisa nije izuzetak. Kao takav, pokazuje solidne performanse u određenim uslovima dok mu se iste urušavaju, na primer, pri povećanju broja učesnika.

U postojećem rešenju, podaci iz tabele tema, organizovani su u vidu liste dok bi isto tako mogli biti u vidu mape sa ključem kao temom. Pristup podacima određenog topika u tom slučaju pokazivao bi bolje performanse.

Takođe, dizajn postojećeg rešenja, pomoću više niti, implicira veću kompleksnost samog rešenja i probleme sinhronizacije nad deljenim podacima i koordinacije između aktivnih niti koje imaju različite uloge. Ukoliko je PubSub servis i sačinjen od više niti, trebalo bi koordinisati redosled izvršavanja između njih, kako bi se izbeglo zauzimanje procesorskog vremena na duže vreme, od strane jedne grupe niti.

Problem predstavljen u ovom radu, čest je problem u računarskim komunikacionim sistemima, stoga je preporuka koristiti standardizovane šablone koji se koriste pri rešavanju ove vrste problema. Jedan od takvih šablona koji bi se mogao primeniti na rešavanje ovog problema je *Reaktor* šablon.